SAND2018-11375R

EXASCALE COMPUTING PROJECT

ECP-U-2018-XXX

Kokkos: Support for ASC Applications and Libraries

WBS STPR 04 Milestone 6

Authors

Christian Trott, Daniel Ibanez, Steven Bova, Nathan Ellingwood, Duane Labreche (1)

Author Affiliations

(1) Sandia National Laboratories

September 19th 2018

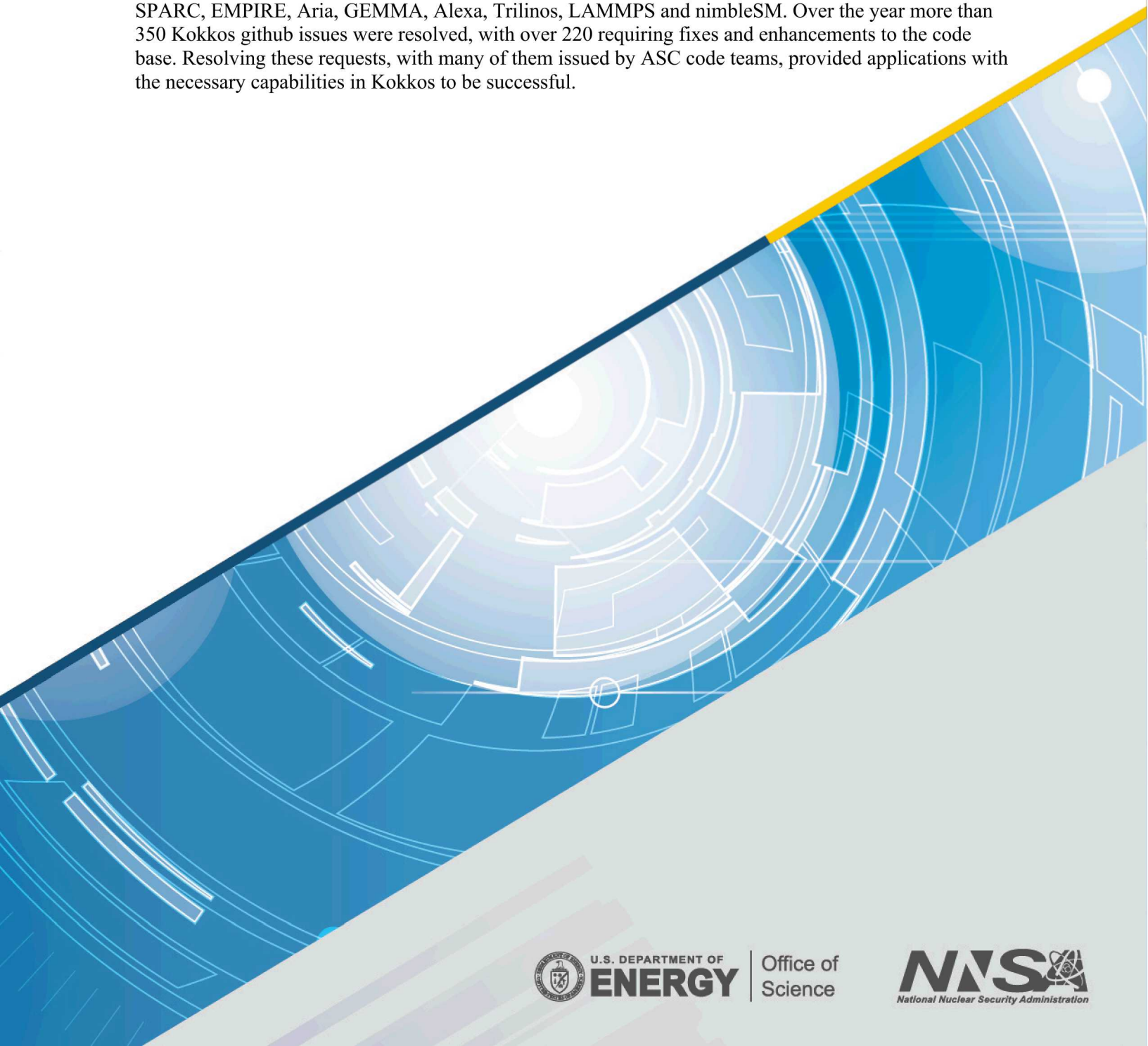
U.S. DEPARTMENT OF ENERGY | Office of Science

National Nuclear Security Administration

EXASCALE COMPUTING PROJECT

EXECUTIVE SUMMARY

This report documents the completion of milestone STPR04-6 Kokkos Support for ASC applications and libraries. The team provided consultation and support for numerous ASC code projects including Sandias SPARC, EMPIRE, Aria, GEMMA, Alexa, Trilinos, LAMMPS and nimbleSM. Over the year more than 350 Kokkos github issues were resolved, with over 220 requiring fixes and enhancements to the code base. Resolving these requests, with many of them issued by ASC code teams, provided applications with the necessary capabilities in Kokkos to be successful.



U.S. DEPARTMENT OF ENERGY | Office of Science

NNSA National Nuclear Security Administration

1. INTRODUCTION

Ongoing support for applications is a necessity to ensure their success in adopting Kokkos and being ready for new supercomputer designs. To do that the Kokkos team must not only provide consultation but also resolve issues disovered by Kokkos adopters in a timely fashion. This report will provide an overview of the amount of work done by the Kokkos team, and will go into details on two examples, one for the development of a new capability, and one for supporting an ASC applications.

2. MILESTONE OVERVIEW

2.1 DESCRIPTION

Help ASC applications and libraries adopt Kokkos by providing consultations on software design, fixing bugs, and making enhancements to Kokkos required by those applications and libraries.

2.2 EXECUTION PLAN

Support for applications and libraries is reactive in nature. It is driven by those applications through feature requests, bug reports and direct support requests. The Kokkos team conducts weekly triage meetings, where issues brought up on github are analysed, prioritized and assigned to developers. The proposed solution approach and time frame for it are then communicated to the customer to see if the plan is acceptable. Simlarily consultation and support requests are discussed in face to face meetings with application teams, and then assigned to specific team members based on availability and best fit.

2.3 COMPLETION CRITERIA

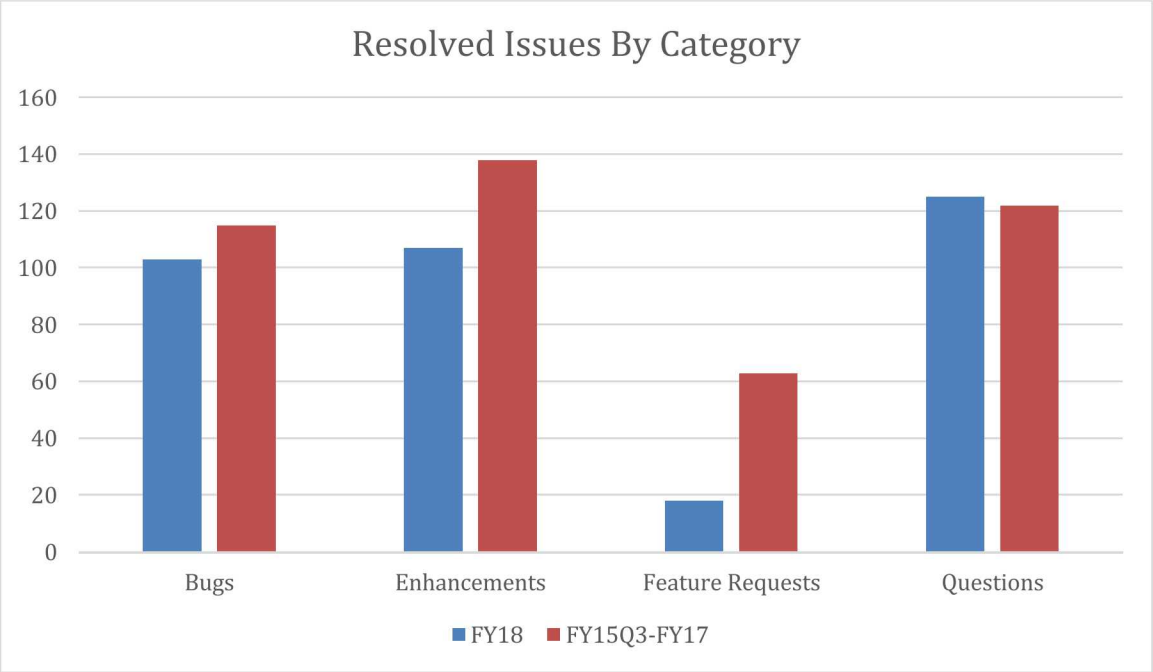
Evidence for executing on support requests will be provided in a report. Application teams are not prevented from reaching their goals by outstanding Kokkos issues.

3. TECHNICAL WORK SCOPE, APPROACH, RESULTS

3.1 APPLICATION SUPPORT AND CONSULTATION

Over the last year application support and consultation has increased further in importance due to a growing number of customers who use Kokkos in more and more parts of their code. That brings on one hand more developers in who have questions, but also exposes an ever increasing number of use cases. Good evidence for that can be obtained from Kokkos' github issue list. The following chart shows the number of resolved issues for FY18 compared to the rest of the time since Kokkos is on github (May 2015 until September 2017). First of all the team resolved almost as many issues in FY18 as in the 2 ½ years before that (353 vs 438). But within those issues the number of full blown "Feature Requests" has been dropping dramatically. "Feature Request" are usually new capabilities while "enhancements" are improvements to existing capabilities, for example to support a previously not supported corner case. This

development is in line with expectations for a maturing project: the basic capabilities applications need are there, but each new code adopting Kokkos exposes nuances in use cases, which require tweaks.



Note that these numbers mean that the Kokkos team on average resolved an issue requiring improvement to the code base on every work day of the past year. We estimate that a little over half of the issues were created by, or on behalf of ASC applications and libraries.

Another aspect of our work is consultations for applications. These took different forms. Some applications have developers matrixed into the Kokkos team, to learn the necessary skills and be intimately familiar with the Kokkos code base. Those developers in turn transfer the knowledge to their application teams and help guide their Kokkos adoption efforts. Other application teams work mostly via opening issues on github and some simply call or go to the office of a Kokkos core developer to ask questions. For Sierra Aria we organized biweekly team room sessions, where we meet in small groups, discuss the porting strategy and debug or profile the code direcly.

3.2 FEATURE EXAMPLE: TILED LAYOUT

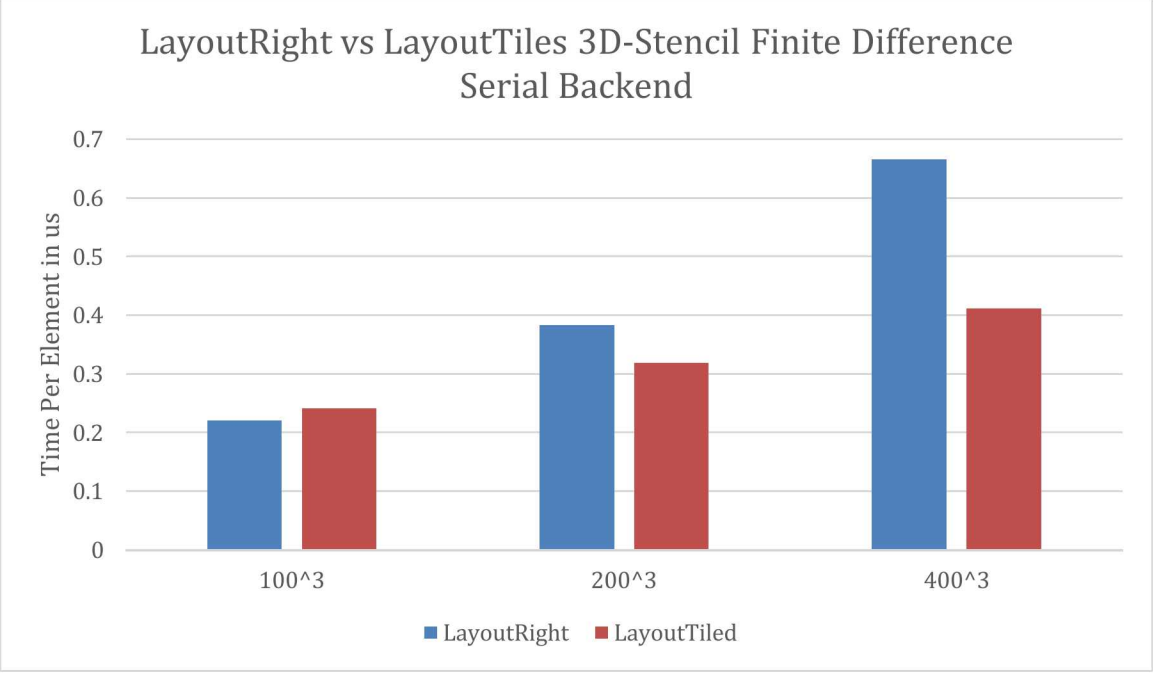
In response to a feature request from the SPARC development team, a new array layout for Kokkos::View was designed and implemented in FY18Q4. SPARC is the next-generation, ATDM-funded application code for simulating atmospheric re-entry flows by approximating the compressible Navier-Stokes equaions using multiple discretizations, and is built on top of Kokkos. Two discretizations that are implemented in SPARC are a structured finite volume method (SFVM) and an unstructured finite volume method (UFVM). The motivation for the request for a new Kokkos array layout is rooted in the fact that the observed performance of the SFVM implemented in SPARC is not as good as that of the UFVM. This observation is contrary to conventional wisdom: it is generally believed that an unstructured algorithm will not be as performant as its structured counterpart.

The accepted explanation for this anomaly lies in the details of the array data structures in SPARC. Fields such as density, momentum, etc are stored in a multidimensional Kokkos::View. For the unstructured code, a two-dimensional view is used, say F(c,v) where c is thefinite volume cell index, which can be

very large ($10^5 - 10^6$) on a single core, and v is the index over the variables such as density, momentum, etc. This latter index is of order 10. The structured mesh code uses a four-dimensional Kokkos::View say F(k,j,i,v), where i,j,k are indices in the three parametric mesh coordinate directions. Assuming similarly sized meshes are in memory in each case, the maximum sizes of i,j,k are roughly the cube root of the maximum size of c. It is believed that the reduced performance of the structured code is associated with the large stride of the access of e.g., F(k+1,j,i,v), which is a stencil neighbor of F(k,j,i,v) but is far away in memory.

To investigate and possibly ameliorate this data access problem, a Kokkos::LayoutTiled was designed and implemented in an attempt to increase the locality of stencil neighbors.

The implementation was done in such a way as to require minimal changes to the application code: the developer need only change the type of layout and Kokkos::MDRangepolicy to tiled. Priliminary results obtained on the CPU in serial demonstrate the improved cache performance for the calculation of a test kernel that computes a stencil-based gradient using a second-order accurate finite difference with four-dimensional Kokkos views similar to those used by SPARC. Tables 1-3 show the timings for the kernel for N = 100, N = 200 and N = 400, respectively for a gradient in each of the coordinate directions. N is the number of cells in each of the i,j,k directions, BC Right is the time to compute the one- sided boundary difference for LayoutRight, BC Tiled is the time for the boundary calculation using the new LayoutTiled, Total Right and Total Tiled are the times for the total kernel using LayoutRight and LayoutTiled, respectively. The data suggests that for small problem sizes the extra overhead associated with the tiled layout may cause sightly degraded performance, but as the problem size gets larger, significant benefits may be realized.



3.3 CASE STUDY: ALEXA RESEARCH CODE

One of the successes of Kokkos' support of ASC applications is the Alexa research code, which is an effort to program shock hydrodynamics from the ground up to run efficiently on Next Generation Platforms (NGP) with a focus on NVIDIA GPUs. Alexa is funded through ASC/IC as part of the broader

ALEGRA effort to support shock hydrodynamics at Sandia. As part of its multi-faceted user support approach, Kokkos provides a variety of examples which are targeted to specific domain areas of High Performance Computing (HPC). One of these examples demonstrates usage of Kokkos to implement the Finite Element Method (FEM). Alexa began as a copy of this example, suitably extended to support the basic equations of motion needed for shock hydrodynamics. This allowed the Alexa team to gain both familiarity with Kokkos and performance on NGP from day one. Several members of the Alexa team had worked extensively on ALEGRA, a successful production ASC/IC application for shock hydrodynamics. The Alexa team was able to move key domain-specific knowledge from ALEGRA into Alexa including time integration methods, shock capturing methods, and material models, all while operating in a codebase which was conformal to high-performance Kokkos usage, ensuring no performance loss due to overheads of legacy code. Kokkos provides a unified API that allows codes to execute on-node parallelism on multi-core CPUs and GPUs. Alexa serves as an example of achieving good performance on both of these architectures without writing any architecture specific code. Using the Kokkos API allows Alexa to compile and perform on several platforms without modifying its source code. Figure 1 shows the scalability of Alexa on several platforms which are relevant to Sandia computing.

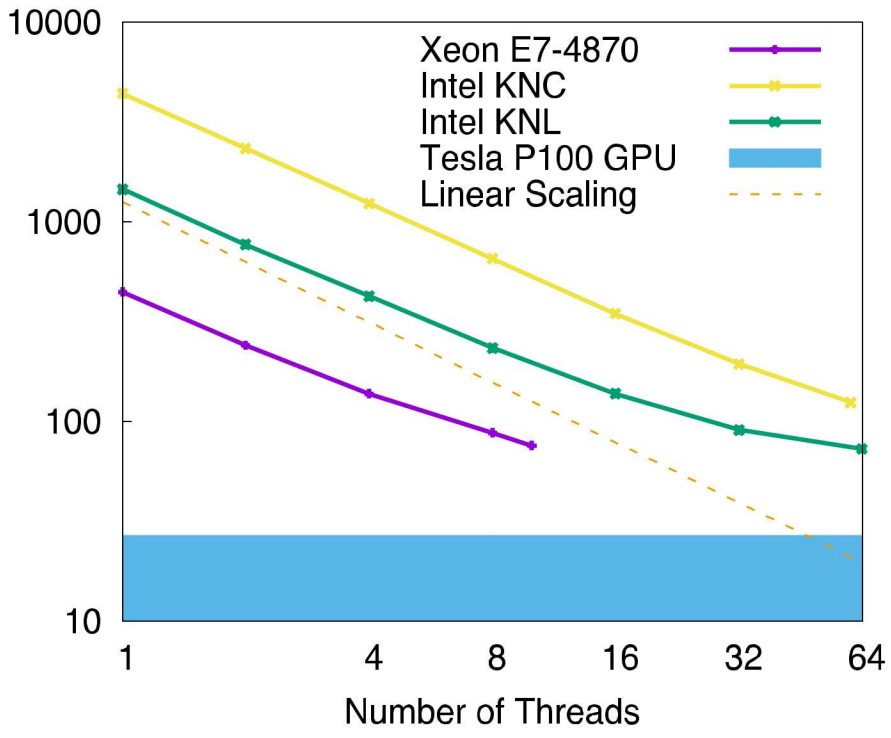


Figure 1: Runtime in seconds over number of threads (GPU has only one data point)

Alexa's goal is to speed up and improve the accuracy of simulations of certain nuclear weapon components. Luckily, there are non-sensitive simulations which we can distribute externally for the purposes of performance evaluation. Through the ALEGRA project's connections with the Department of Defense (DoD), there was increased interest at the DoD to evaluate performance of Alexa on a newly-acquired cluster, Hokule'a. This cluster was heavily powered by NVIDIA GPUs, motivating the use of an NGP-performant code like Alexa. The DoD funded a contractor to carry out a performance study of

Alexa on the Hokule'a machine, which produced positive results both in terms of scalability and the ease of compilation for different architectures (the latter being due to Kokkos' build system capabilities which isolate the user from hardware and compiler details). Figure 2 shows the result of one of the scaling studies conducted on Hokule'a.

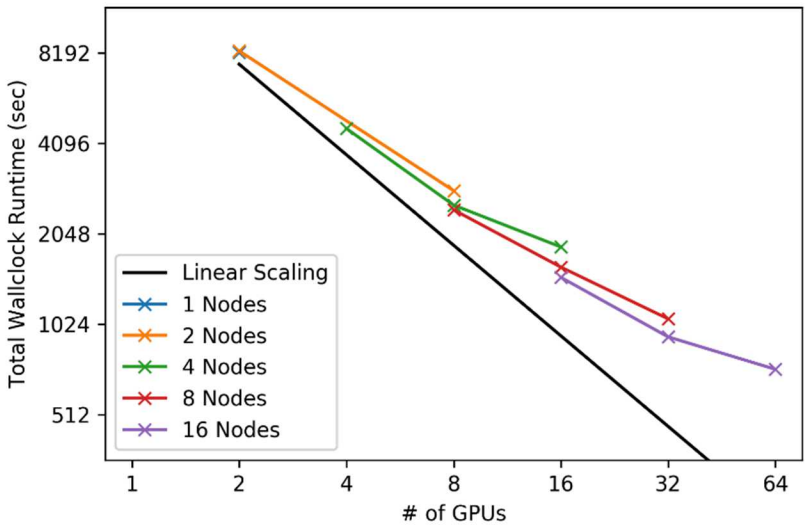


Figure 2: Scaling of Alexa using many GPUs on the DoD Hokule'a cluster

3.4 RESOURCE ACCESS

For evidence and more details on the feature issues resolved and enhancement made to Kokkos visit the github repository issue list and see the changelog.

Feature	Location	Access Restriction
Kokkos Issues	https://github.com/kokkos/kokkos/issues	Open access.
Kokkos Changelog	https://github.com/kokkos/kokkos/blob/master/CHANGELOG.md	Open access.

4. RESOURCE REQUIREMENTS

The work performed here required 1.75 FTE.

5. CONCLUSIONS AND FUTURE WORK

As support for Kokkos users is an ongoing effort, the team must maintain this level of effort. It will continue to resolve issues brought by customers and engage the application teams to ensure their success in adopting Kokkos.

6. ACKNOWLEDGMENTS

This research was supported by the Exascale Computing Project (ECP), Project Number: 17-SC-20-SC, a collaborative effort of two DOE organizations—the Office of Science and the National Nuclear Security Administration—responsible for the planning and preparation of a capable exascale ecosystem—including software, applications, hardware, advanced system engineering, and early testbed platforms—to support the nation's exascale computing imperative.

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology and Engineering Solutions of Sandia, LLC., a wholly owned subsidiary of Honeywell International, Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA-0003525.